

# Replacing Cross-Validation with Interrogation: A Universal Test for Underfitting and Overfitting

THIS VERSION: April 23, 2025

**Megan Czasonis**

Managing Director

State Street Associates

[mczasonis@statestreet.com](mailto:mczasonis@statestreet.com)

**Yin Li**

Assistant Vice President

State Street Associates

[yli31@statestreet.com](mailto:yli31@statestreet.com)

**Huili Song**

Vice President

State Street Associates

[hsong3@statestreet.com](mailto:hsong3@statestreet.com)

**David Turkington**

Senior Managing Director

State Street Associates

[dturkington@statestreet.com](mailto:dturkington@statestreet.com)

## Abstract

Gauging the reliability of a prediction routine for use outside its training data is a fundamental challenge in machine learning. Model training typically relies on cross-validation to avoid overfitting, whereby alternate calibrations of a model are trained on subsamples of the available data and tested on the corresponding validation samples. However, cross-validation has inherent limitations: it cannot directly evaluate the model trained on all available data, sample slicing limits the statistical power of subsample training and testing, and it is computationally expensive. We propose an alternative approach based on interrogating the primary model trained on all available data. In our simulations, the interrogation-based method successfully identified near-optimal model calibrations without using any validation samples. Our model-agnostic method works by explicitly decomposing prediction logic into linear, nonlinear, pairwise, and high-order interaction components, and then testing for the statistical identifiability of these separate components in the presence of noise. Poorly calibrated models reveal statistical problems analogous to harmful collinearity in linear regression. Interrogation can be applied in a wide variety of contexts to evaluate model reliability.

# Replacing Cross-Validation with Interrogation: A Universal Test for Underfitting and Overfitting

Gauging the reliability of a prediction routine for use outside its training data is a fundamental challenge in machine learning. Prediction functions can be underfitted, failing to capture important patterns in the data. Alternatively, they can be overfitted, reflecting spurious relationships and noise in the training sample. Both cases suggest that a learned prediction function is suboptimally calibrated and unlikely to perform well on unseen data. The question then becomes how to evaluate calibration quality using only the training sample.

The conventional approach to this problem is to withhold a portion of the available training data as a validation sample and use it to evaluate the performance of a prediction function trained on the complementary sample. When this process is repeated for multiple validation samples it is called cross-validation. Validation results are used to select hyperparameters for model training or to otherwise guide the selection of a prediction function that is expected to perform well on unseen data. This process essentially simulates the application of a prediction routine to unseen data within the training sample by carving it into smaller subsamples.

There are multiple disadvantages to cross-validation. First, it cannot directly evaluate a prediction function trained on all available data, which in principle is the ideal way to train a model for future use. Cross-validation must evaluate models trained on smaller sets of data because some of the data is withheld for validation. The retrained subset models do not necessarily reflect the full-data model faithfully. Second, each subsample split is subject to noise in training and noise in testing due to relatively smaller samples. Third, repeated retraining is computationally expensive. For example, we could try to minimize the previous two disadvantages by withholding validation samples of a single observation, training on the remaining  $N - 1$  observations. Repeating this experiment for a sufficiently large number of validation points may be prohibitive. Fourth, retraining requires full knowledge of the prediction model and its training regimen. The validation approach cannot evaluate alternative models if they are opaque.

In this paper, we introduce a test for underfitting and overfitting that overcomes the limitations of cross-validation noted above. Our approach is based on interrogation. For a given prediction function, we:

1. Form predictions for a range of hypothetical tasks.
2. Compute partial predictions for linear, nonlinear, pairwise interaction, and high-order interaction components.
3. Estimate linear regression betas for  $Y$  outcomes regressed on these four components.
4. Test for the joint hypothesis that all four betas are equal to 1.

Intuitively, this test evaluates whether four logical components that are measured to be conceptually distinct are in fact statistically distinct and identifiable in a linear regression. Poorly calibrated models often have logical components that are badly scaled or that address redundant effects (positively correlated components) or cancel each other out (negatively correlated components). Interrogation can be applied to black-box prediction functions to compare their logical robustness. It can also be used to determine an effective stopping time for learning during training. Overall, it shows how model-agnostic transparency into the logic of a prediction routine provides useful foresight into its efficacy beyond the training sample.

We perform simulation tests on neural network models capable of overfitting empirical samples. We consider a range of data-generating processes with different levels of underlying complexity. Our results show that interrogation consistently selects prediction functions with near-optimal performance in subsequent testing samples, thus avoiding underfitted and overfitted calibrations. Across different data-generating processes, interrogation selects prediction functions that differ in the number of training epochs and differ in the contribution of each logical component, which suggests that the methodology cannot be approximated by a simple heuristic.

We proceed as follows. First, we describe the interrogation methodology in detail. Next, we describe the setup of our simulation tests in terms of underlying assumptions and illustrative model training. We then present the simulation results, including a comparison to traditional cross-validation. In the following section, we apply the same evaluation framework to real-world data for the case of predicting currency prices. Lastly, we provide some additional discussion regarding potential extensions of this methodology, and we end with a conclusion.

## Interrogation Methodology

Assume there exists a prediction function,  $\hat{y} = \hat{f}(x_1, x_2, \dots, x_M)$  for  $M$  feature inputs. We do not need to know the mechanics of  $\hat{f}$ , just that it yields a deterministic prediction for any set of inputs.

### Model Fingerprint

We apply the Model Fingerprint method of Li, Turkington, and Yazdani (2020) to measure the distinct prediction effects of  $\hat{f}$ . First, we select one of the features,  $x_k$ , and for one of its values  $x_{ki}$  for observation  $i$  in a data sample, we compute the partial prediction following Friedman (2001) as the expected value of the prediction when  $x_k = x_{ki}$ , marginalized over all other values of the other features. Empirically, the partial prediction  $\hat{f}_k$  for input  $x_{ki}$  is the average:

$$\hat{f}_k(x_{ki}) = \frac{1}{N} \sum_{j=1}^N \hat{f}(x_{1j}, \dots, x_{ki}, \dots, x_{Mj}) \quad (1)$$

We repeat Equation 1 for  $k = 1, \dots, M$  and  $i = 1, \dots, N$  to obtain partial prediction functions conditioning on one feature at a time. In practice, it is possible to estimate the partial prediction functions using sparse samples of observations, but here we describe the process assuming we use all  $N$  observations. Note that for the special case of linear relationships, the partial prediction functions will equal the product of the linear regression beta coefficient and the input  $x_{ki}$ . More generally, though, the partial prediction functions will trace nonlinear relationships. We decompose input  $x_k$ 's partial prediction function into a linear component and a nonlinear component by obtaining the best fit (least squares) regression line for the function using all the sampled observations. Now, for every input in the training sample we can estimate a linear component and a nonlinear component for each predictive input.

Next, we compute the pairwise interaction effect for a pair of features  $x_k$  and  $x_l$  beyond the independent effect of the features. Specifically, the incremental pairwise interaction effect of  $x_k = x_{ki}$  and  $x_l = x_{li}$  is the partial prediction  $\hat{f}_{k,l}(x_{ki}, x_{li})$  minus the individual prediction effects:

$$\hat{f}_{k,l}(x_{ki}, x_{li}) = \frac{1}{N} \sum_{j=1}^N \hat{f}(x_{1j}, \dots, x_{ki}, x_{li}, \dots, x_{Mj}) - \hat{f}_k(x_{ki}) - \hat{f}_l(x_{li}) \quad (2)$$

We repeat Equation 2 for every pair  $(k, l)$  and  $i = 1, \dots, N$  to obtain pairwise interaction effects that are distinct and incremental to the individual input effects. Note that for the special case of linear relationships, pairwise interaction effects will equal precisely zero. More generally, pairwise interactions reflect important conditional effects of a prediction function.

Now that we have linear, nonlinear, and pairwise interaction effects, we simply define high-order interaction effects as any remainder of  $\hat{y}$  that is not explained by all the other measured effects. This decomposition has the virtue that we are not required to compute high-order effects explicitly, as doing so becomes increasingly prohibitive in terms of calculation time.

Next, for all  $N$  observations in the available sample, we aggregate effects to arrive at composite linear, nonlinear, pairwise interaction, and high-order interaction effects:

$$\hat{f}_{linear}(x_i) = \sum_{k=1}^M \hat{f}_{k,linear}(x_{ki}) \quad (3)$$

$$\hat{f}_{nonlinear}(x_i) = \sum_{k=1}^M \hat{f}_{k,nonlinear}(x_{ki}) \quad (4)$$

$$\hat{f}_{pairwise}(x_i) = \sum_{k=1}^M \sum_{l=1}^M \hat{f}_{k,l}(x_{ki}, x_{li}) \quad (5)$$

$$\hat{f}_{highorder}(x_i) = \hat{f}(x_i) - \hat{f}_{linear}(x_i) - \hat{f}_{nonlinear}(x_i) - \hat{f}_{pairwise}(x_i) \quad (6)$$

This procedure exactly decomposes every prediction into four high-level logical components which form the basis of the test we describe in the following section. Before we proceed, it is worth noting that although we have assumed that the training sample is used to compute the logical components, it is possible to compute the prediction components using any sample of prediction tasks. Also, while we have chosen to aggregate the prediction components into four high-level categories, it is possible to aggregate them differently, or not at all.

## Test for Underfitting and Overfitting

We propose a diagnostic test for underfitting and overfitting based on the four components from Equations 3 through 6 together with the corresponding outcomes for  $Y$ . In particular, we estimate the linear regression betas for  $Y$  regressed on the four prediction components.

$$y = \alpha + \beta_1 \hat{y}_{linear} + \beta_2 \hat{y}_{nonlinear} + \beta_3 \hat{y}_{pairwise} + \beta_4 \hat{y}_{highorder} + \eta \quad (7)$$

Our null hypothesis is that for a well-calibrated prediction function, each of the four betas equals 1,  $H_0: \beta_1 = \beta_2 = \beta_3 = \beta_4 = 1$ .

Let us take a moment to consider the intuition behind this test. We know that the four components comprise an exact decomposition of the predictions, so a regression of  $\hat{y}$  on the components would yield coefficients trivially equal to 1. The regression we perform is different because it includes the predictive error terms,  $\varepsilon = y - \hat{y}$ . We are interested to see if the prediction components can be identified with statistical precision in the presence of empirical errors. A poorly calibrated prediction function will likely struggle to recover its distinct components when faced with noise, even if the observations correspond to the sample that was used for training. This ambiguity is a sign of instability in the prediction function.

Our statistical test is a conventional  $F$ -statistic for the null hypothesis that all four betas equal 1. We compute the  $p$ -value of that  $F$ -statistic which corresponds to an intuitive interpretation of whether to reject the null hypothesis that a prediction function is well-calibrated. It is worth noting that in practice, the  $F$ -statistic is often closely aligned to the sum of squared  $t$ -statistics for the betas (it is not equivalent unless the four components are completely orthogonal). Therefore, it can be instructive to evaluate the  $t$ -statistics of the four components to gain further insight into their behavior individually.

## Simulation Test Setup

We evaluate the efficacy of our proposed interrogation test to select a stopping time for neural network training. We run simulation experiments for eight data-generating processes with different amounts of true complexity. Following standard convention, we initialize neural network parameters randomly and train them in successive epochs, where each epoch constitutes one pass through  $N$  training observations using gradient descent and back-propagation to update learned parameter values. With enough epochs, training error (mean squared error of predictions evaluated on the training sample) continually decreases and converges towards zero. However, testing error (mean squared error of predictions evaluated on the testing sample) eventually rises due to overfitting. We apply interrogation to the learned prediction function after each epoch, which allows us to select an ex-ante optimal stopping time without the use of any heuristics or subsample cross-validation tests.

## Data Generation

We perform eight separate simulations using different data-generating processes. Our goal is to create different types of nonlinear complexity in a reasonably straightforward way. Each simulation shares the following setup:

- We generate 1000 observations from Regime 1 (dominant regime) and 500 observations from Regime 2 (secondary regime).
- The input values for  $X_1$  and  $X_2$ , representing Regime 1 and Regime 2, respectively, contain 10 feature columns. We draw values for each observation for each feature independently from a standard normal distribution,  $N(0,1)$ .
- To reflect noise in outcomes, we draw values for  $\varepsilon_1$  and  $\varepsilon_2$ , representing Regime 1 and Regime 2, respectively, from a standard normal distribution,  $N(0,1)$ .
- If a regime's data-generating process relies on parameters for linear weights vectors,  $b_1$  and  $b_2$ , or conditionality matrices,  $C_1$  and  $C_2$ , these parameter values are drawn from a standard normal distribution,  $N(0,1)$ .

Exhibit 1 summarizes the high-level design properties of each simulation.

Exhibit 1: Summary of simulation design

Simulation	Regime 1	Regime 2	Pairwise interaction	High-order interaction
A	Linear	Linear		
B	Linear	Nonlinear		
C	Nonlinear	Linear		
D	Linear	Nonlinear	Yes	
E	Linear	Nonlinear	Yes	
F	Nonlinear	Linear	Yes	
G	Linear	Nonlinear	Yes	Yes
H	Nonlinear	Linear	Yes	Yes

The following equations provide the simulation-specific data-generating details for each simulation.

Simulation A: Linear models within regimes

$$y_1 = X_1 \cdot b_1 + 0.1\varepsilon_1 \quad (8)$$

$$y_2 = X_2 \cdot b_2 + 0.1\varepsilon_2 \quad (9)$$

Simulation B: Introduce nonlinear effect in secondary regime

$$y_1 = X_1 \cdot b_1 + 0.1\varepsilon_1 \quad (10)$$

$$y_2 = \sin(X_2) \cdot b_2 + 0.1\varepsilon_2 \quad (11)$$

Simulation C: Nonlinear effect in dominant regime

$$y_1 = \sin(X_1) \cdot b_1 + 0.1\varepsilon_1 \quad (12)$$

$$y_2 = X_2 \cdot b_2 + 0.1\varepsilon_2 \quad (13)$$

Simulation D: Introduce pairwise interactions with additional nonlinearity in secondary regime

$$y_1 = X_1 \cdot b_1 + \sum_{i=1}^{10} X_1 C_1 \cdot x_i + 0.1\varepsilon_1 \quad (14)$$

$$y_2 = \sin(X_2) \cdot b_2 + \sum_{i=1}^{10} \sin(X_2 C_2) \cdot \cos(x_i) + 0.1\varepsilon_2 \quad (15)$$

Simulation E: Increase random noise

$$y_1 = X_1 \cdot b_1 + \sum_{i=1}^{10} X_1 C_1 \cdot x_i + 0.2\varepsilon_1 \quad (16)$$

$$y_2 = \sin(X_2) \cdot b_2 + \sum_{i=1}^{10} \sin(X_2 C_2) \cdot \cos(x_i) + 0.2\varepsilon_2 \quad (17)$$

Simulation F: Reverse the dominant and secondary regimes from simulation D

$$y_1 = \sin(X_1) \cdot b_1 + \sum_{i=1}^{10} \sin(X_1 C_1) \cdot \cos(x_i) + 0.1\varepsilon_1 \quad (18)$$

$$y_2 = X_2 \cdot b_2 + \sum_{i=1}^{10} X_2 C_2 \cdot x_i + 0.1\varepsilon_2 \quad (19)$$

Simulation G: Introduce explicit high-order interactions (described below)

$$y_1 = X_1 \cdot b_1 + \sum_{i=1}^{10} X_1 C_1 \cdot x_i + HI_{l,1} + HI_{l,2} + 0.1\varepsilon_1 \quad (20)$$

$$y_2 = \sin(X_2) \cdot b_2 + \sum_{i=1}^{10} \sin(X_2 C_2) \cdot \cos(x_i) + HI_{n,1} + HI_{n,2} + 0.1\varepsilon_2 \quad (21)$$

Simulation H: Reverse the dominant and secondary regimes from simulation G

$$y_1 = \sin(X_1) \cdot b_1 + \sum_{i=1}^{10} \sin(X_1 C_1) \cdot \cos(x_i) + HI_{n,1} + HI_{n,2} + 0.1\varepsilon_1 \quad (22)$$

$$y_2 = X_2 \cdot b_2 + \sum_{i=1}^{10} X_2 C_2 \cdot x_i + HI_{l,1} + HI_{l,2} + 0.1\varepsilon_2 \quad (23)$$

We define the high-order interaction ( $HI$ ) terms based on randomly selected subsets of features,  $S_j$ . For each  $HI$  term, we randomly select at least three features, so  $|S_j| \geq 3$ . The interaction term is then computed as the element-wise product of the selected feature values.

For linear high-order interactions,  $HI_{l,j} = \prod_{k \in S_j} X_{l,k}$ .

For nonlinear high-order interactions,  $HI_{n,j} = \prod_{k \in S_j} \sin(X_{n,k})$ .

## Sample Definitions

All observations are sampled independently, so there are no dependencies between them. For each simulation, we randomly select 80 percent (1200 observations) from the total sample of  $N = 1500$  observations for the training sample, and we set aside the remaining 20 percent (300 observations) as a testing sample. All model calibration, as well as interrogation, is performed exclusively on each simulation's training sample.

## Model and Training

For each simulation, we train a neural network model with five fully connected layers of 100 neurons each. We choose this architecture because it is relatively simple yet still capable of fitting, and overfitting, the training samples. Each hidden neuron applies a ReLU (rectified linear unit) activation function.

The model is trained on all 1200 available observations in the training sample. Training uses the standard approach of back-propagation with gradient descent. We use a learning rate of 0.0001 and use a batch size equal to the size of the training sample.

## Simulation Test Results

We now present the results of interrogation applied to the simulations described in the previous section.

### Variance Contribution of Predictive Components

Recall that a given simulation has multiple training epochs, and each epoch corresponds to its own prediction function. We begin by computing the four predictive components from Equations 3 through 6 for every prediction function, using the following decomposition of  $\hat{y}$  versus its average value,  $\mathbb{E}[\hat{y}]$ :

$$\hat{y} - \mathbb{E}[\hat{y}] = \hat{y}_{linear} + \hat{y}_{nonlinear} + \hat{y}_{pairwise} + \hat{y}_{highorder} \quad (24)$$

This decomposition gives us insight into the nature of the learned function. Before we proceed to the quality of calibration test, it is interesting to observe the relative importance of each predictive component across different simulations and epochs. We compute a variance decomposition of the components for a specific prediction function as:

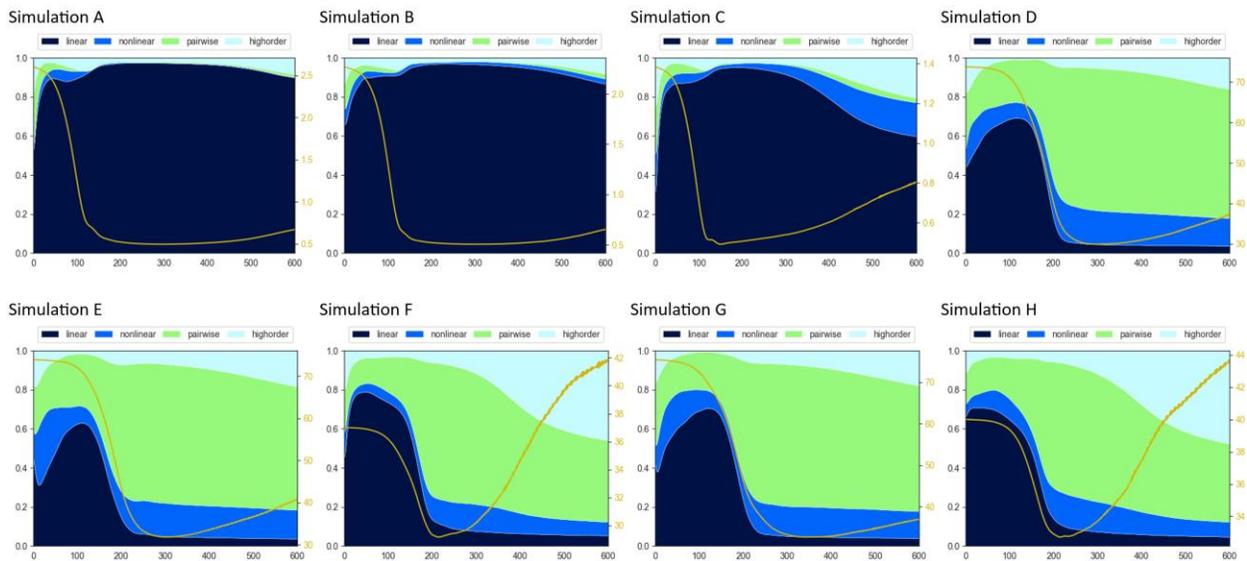
$$v_i = \frac{\sum_j \text{Cov}(\hat{y}_i, \hat{y}_j)}{\sum_{i,j} \text{Cov}(\hat{y}_i, \hat{y}_j)} \quad (25)$$

Exhibit 2 shows the fraction of variance over training epochs for each simulation. The overlaid yellow line shows the mean squared error (MSE) evaluated on the withheld testing sample for

each prediction function. The prediction function that coincides with the lowest MSE performed the best in the testing sample, but it is important to remember that we do not know the yellow line in advance. We show it here to indicate for intuition, ex-post, which prediction functions appeared to be well-calibrated for the testing sample.

There are two important points to note about Exhibit 2. First, the ex-post optimal prediction functions in these simulations are driven by different components. The predictions for the relatively simpler processes A, B, and C, are mostly driven by linear effects, whereas pairwise interactions become important in the others. Second, the ex-post optimal number of epochs varies substantially across simulations.

Exhibit 2: Fractional Contribution to Prediction Variance

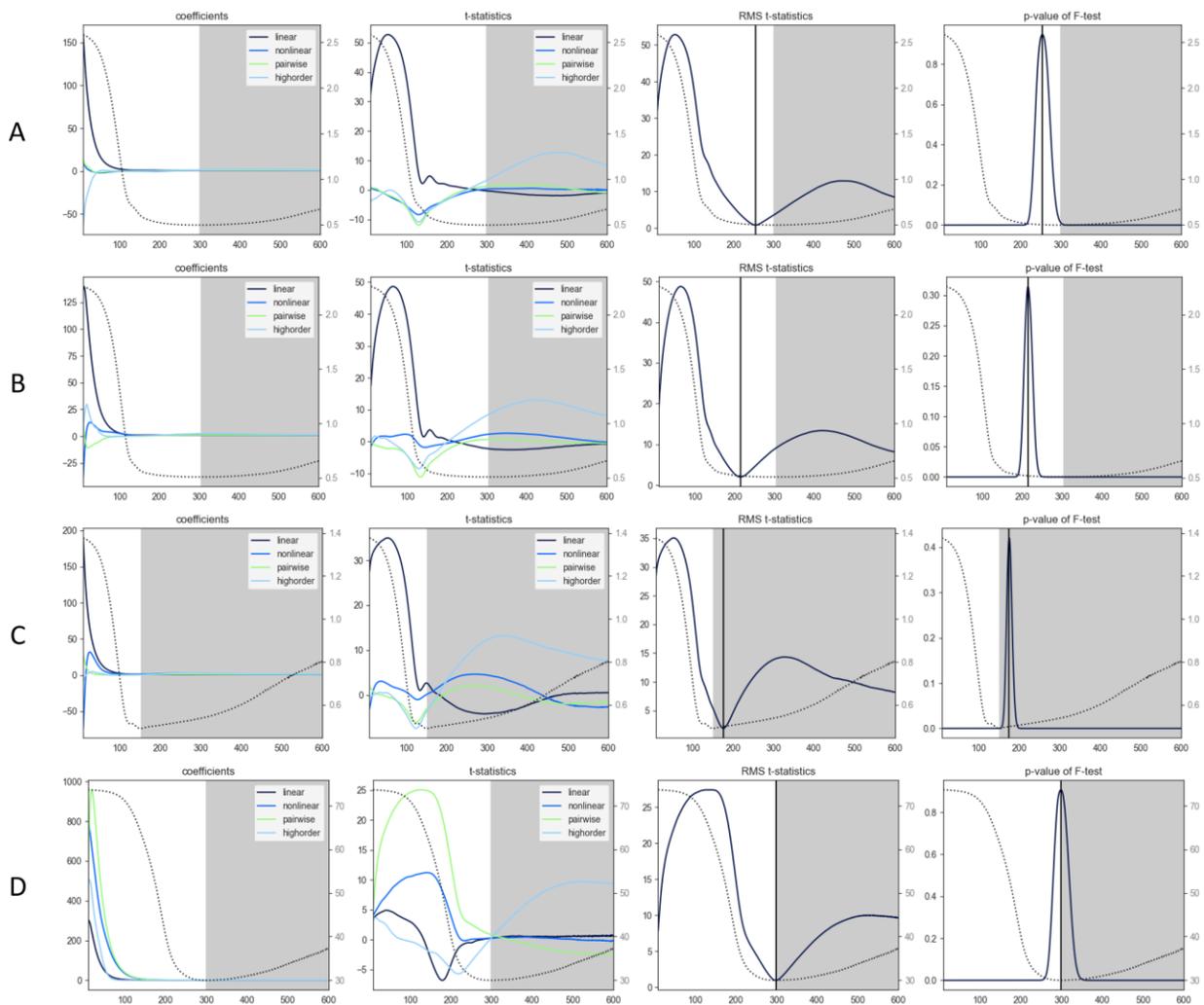


### Interrogation Test for Underfitting and Overfitting

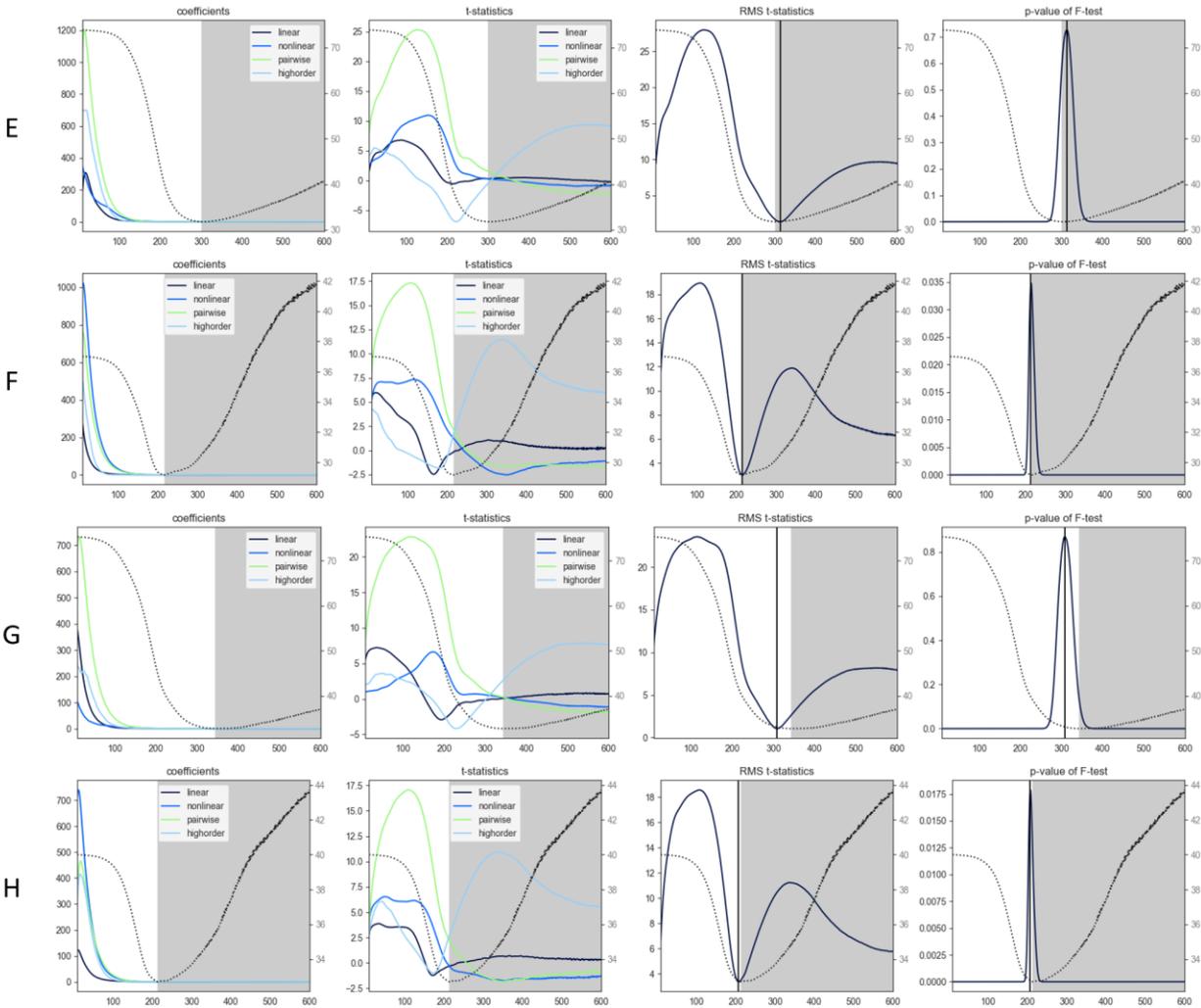
We regress the  $Y$  outcomes from each training sample on the corresponding prediction components from those same training samples, and we conduct the statistical test described earlier to evaluate the hypothesis that all coefficients from the linear regression equal 1. Exhibit 3 shows these results for each simulation. The leftmost panel plots the beta coefficients. The next panels show the  $t$ -statistics of the betas and the root mean squared (RMS)  $t$ -statistic, respectively. A vertical black line denotes the minimum value of the RMS  $t$ -statistic. The rightmost panel shows the  $p$ -value of the  $F$ -test for the interrogation test joint hypothesis that all betas equal 1. The dark gray background shading begins when the mean squared error (MSE) evaluated on the withheld testing sample, as shown by the dotted line, rises above its global minimum. Therefore, the dark gray shaded region suggests overfitting.

Exhibit 3 reveals that the interrogation test chooses an ex-ante stopping point very close to the ex-post optimal stopping point indicated by the testing sample. It should be noted that the testing samples also contain noise, so their optimal stopping points are not perfectly conclusive. Nevertheless, the near-optimality of the interrogation test selections compared to the ex-post testing sample outcomes, without the need for any subsample cross-validation exercises, is an encouraging result. In addition, it is interesting to note the variation in patterns for the individual  $t$ -statistics, and to observe the volatility of the RMS  $t$ -statistic value in the realm of overfitting.

Exhibit 3: Test for Underfitting and Overfitting



### Exhibit 3 (continued)

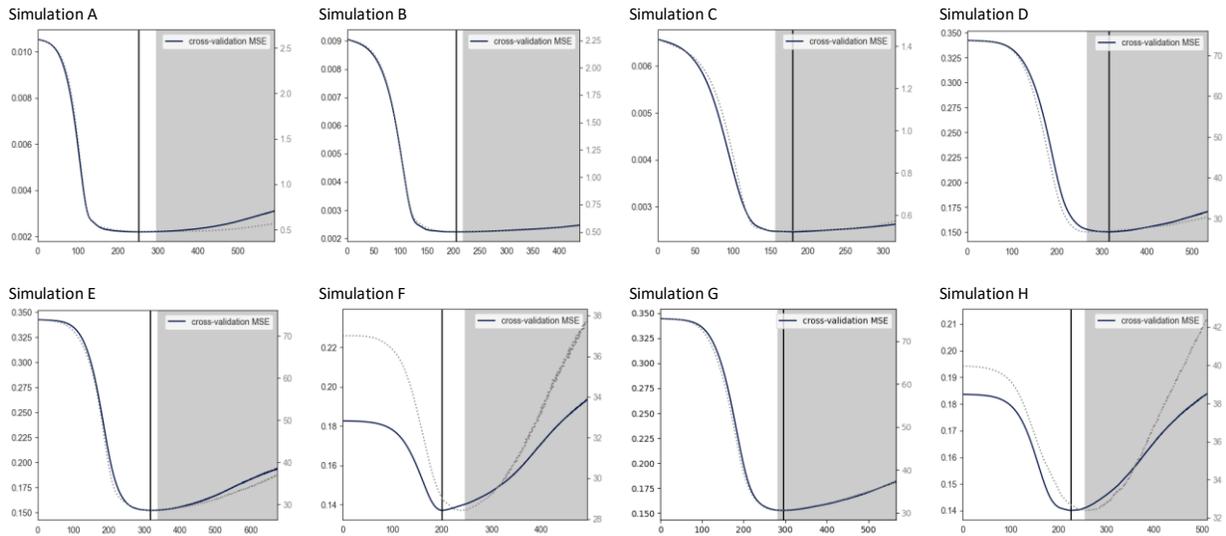


### Comparison to Cross-validation

For comparison, we evaluate a traditional five-fold cross-validation scheme. To do so, we partition the training sample into five equally sized subsets (folds). In each iteration, one fold is set aside as the validation sample, while the remaining four folds are used for training. This process is repeated five times, each time using a different fold for validation. We compute the MSE on the validation fold in each iteration and obtain the final cross-validation error by averaging the validation MSE across all five iterations. The stopping point for training is selected as the epoch for which the cross-validation error reaches its minimum.

We compare the selected stopping point to the epoch corresponding to the actual minimum MSE observed on the separate testing sample. The results indicate that cross-validation does not necessarily yield the ex-post best performing model, especially when the data-generating process is complex.

## Exhibit 4: Cross-validation versus Testing Error



## Empirical Test Results

We apply the same evaluation framework to real-world data for the case of predicting currency prices. Our goal is to assess the usefulness of the interrogation test for models trained on empirical data, not to design the best currency prediction model. As such, the set-up is intentionally simple.

### Data

We predict 20-day returns for exchange rate pairs from 10 major currencies: Australian dollar (AUD), Canadian dollar (CAD), Swiss franc (CHF), Euro (EUR), British pound (GBP), Japanese yen (JPY), Norwegian krone (NOK), New Zealand dollar (NZD), Swedish krona (SEK), and US dollar (USD). For each pair of currencies, we consider the exchange rate quoted in both directions (for example, AUD per CAD and CAD per AUD), for a total of 90 exchange rates in our sample.

For each exchange rate pair, we use six predictor variables:

- Short-term interest rate differential – Difference in one-month interest rates between the two countries
- Long-term equity differential – Difference in trailing 11-month equity returns (ending one-month prior) between the two countries
- Short-term equity differential – Difference in trailing one-month equity returns between the two countries
- Trend – Trailing 12-month return of the given exchange rate
- Valuation – Deviation of the given pair’s real exchange rate (implied by OECD PPP) from fair value (estimated as the trailing five-year average real exchange rate)
- Investor flow differential – Difference in the cross-sectional percent rank of trailing 20-day average flows between the two currencies<sup>1</sup>

Our full dataset consists of daily observations for 90 exchange rate pairs over the period January 1, 2013 through December 31, 2018. We use the first five years of data (2013 through 2017) as the training sample and the last year of data (2018) as the testing sample.

## **Model and Training**

We train a neural network model with the same architecture as in the simulation tests: five fully connected layers of 100 neurons each and a ReLU activation function.

The model is trained on all observations in the training sample (2013 through 2017) which we structure as a panel dataset. The panel set-up allows the model to be trained on increased variability from a greater number of observations. We use a learning rate of 0.001 and a batch size equal to the size of the training sample.

## **Results**

Exhibit 5 shows the variance contribution of predictive components over training epochs for the currency prediction functions. It reveals that the ex-post optimal prediction function is mostly driven by linear and nonlinear effects, albeit with some high order interactions. This suggests some complexity in the relationships learned by the model. Compared to the simulation-based results in Exhibit 2, the empirical relationships appear more complex than processes A through C, but less complex than processes D through H.

Exhibit 5: Fractional Contribution to Prediction Variance – Empirical

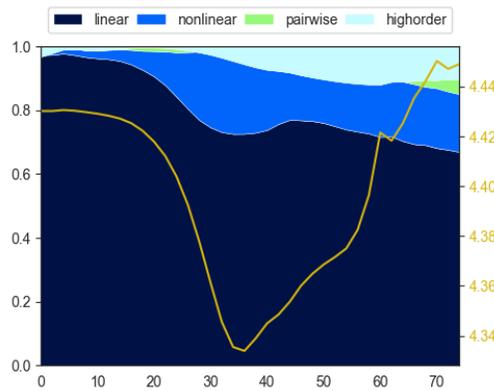
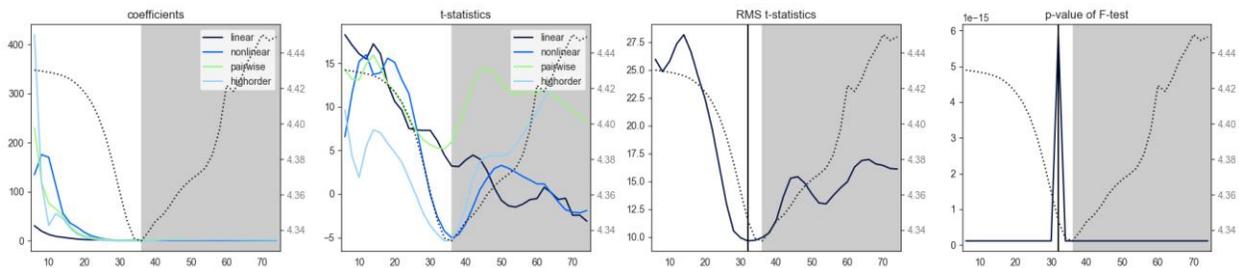


Exhibit 6 shows the results of the interrogation tests on the currency prediction functions. The individual  $t$ -statistics are more volatile across epochs than in the simulation tests, which is unsurprising given the empirical data likely contains more noise than the simulated processes. Nonetheless, the RMS  $t$ -statistic and F-tests still choose a clear ex-ante stopping point that is very close to the ex-post optimal stopping point indicated by the testing sample. This is an encouraging result, suggesting that the interrogation test may be useful for detecting overfitting and underfitting even in real-world cases with low signal to noise.

Exhibit 6: Test for Underfitting and Overfitting – Empirical



## Discussion and Potential Extensions

One of the key benefits of the interrogation approach is its generality. It can be used to compare prediction functions even when the functions are opaque and are not possible to retrain. It can also be used when we do not know which data was included in the training sample. We can implement the process using a different sample of observations for  $X$  and  $Y$ . In the examples above, we showed the efficacy of interrogation based on observations from the training sample. Therefore, possible overlap with training sample observations is not a cause for concern.

In addition, we suspect it is possible to benefit from versions of the interrogation diagnostic test applied to more granular prediction components than the four we have outlined. The Model Fingerprint method of accounting for logical components gives  $M$  individual linear components,  $M$  nonlinear components,  $M(M - 1)$  pairwise interactions, and any number of desired higher-order components such as three-way interactions, four-way interactions, and so forth. In principle, the interrogation approach can be applied to any subset of these components.

Lastly, we suspect that it may be possible to mitigate some negative artifacts of overfitting even when faced with pre-trained black-box models by extracting information from the predictive subcomponents and exploiting features of their relationships to each other.

We leave these extensions to future research.

## Conclusion

We introduced a novel test for detecting underfitting and overfitting that does not require the use of validation samples. Our approach is completely general and relies on a process of interrogation, whereby we use responses from a prediction function to isolate distinct logical components. The test then uses linear regression to observe whether these components can be clearly identified in the presence of additional noise. The lack of clear identification indicates likely underfitting or overfitting of a prediction function.

We evaluated the interrogation routine by training neural network models on eight simulated samples of data-generating processes with varying degrees of underlying complexity. The test reliably selected effective ex-ante stopping times for neural network training, corresponding to prediction functions that performed nearly as well as the ex-post optimal testing sample error, and often better than traditional cross-validation. We also found that the interrogation test selected an effective ex-ante stopping point when applied to models trained on empirical data for the case of predicting currency prices.

## Notes

This material is for informational purposes only. The views expressed in this material are the views of the authors, are provided “as-is” at the time of first publication, are not intended for distribution to any person or entity in any jurisdiction where such distribution or use would be contrary to applicable law and are not an offer or solicitation to buy or sell securities or any product. The views expressed do not necessarily represent the views State Street Global Markets® or State Street Corporation® and its affiliates.

## References

Friedman, J. H. 2001. “Greedy Function Approximation: A Gradient Boosting Machine.” *Annals of Statistics*, 29 (5).

Li, Y., D. Turkington and A. Yazdani. 2020. “Beyond the Black Box: An Intuitive Approach to Investment Prediction with Machine Learning. *The Journal of Financial Data Science*, 2 (1).

---

<sup>1</sup> Flows are measured as the net buying or selling of foreign exchange forwards, as captured by State Street’s Foreign Exchange Flow Indicators.